AMR Annotation:
Special Topics

http://tiny.cc/amrtutorial

AMR

Coordination & Clausal Connectives

Coordination

- We invited the psychics, JFK, and Stalin.

```
(i / invite-01
   :ARG0 (w / we)
   :ARG1 (a / and
            :op1 (p / person :mod (p4 / psychic))
            :op2 (p2 / person :name (n / name :op1 "JFK"))
            :op3 (p3 / person :name (n2 / name :op1 “Stalin"))))
```

- Our invitees: the psychics, JFK, Stalin. (*implicit*)

Coordination

- We invited the psychics, JFK and Stalin.

```
(i / invite-01
   :ARG0 (w / we)
   :ARG1 (a / and
            :op1 (p / person
                    :mod (p3 / psychic)
                    :name (n / name :op1 "JFK"))
            :op2 (p2 / person
                    :mod (p4 / psychic)
                    :name (n2 / name :op1 “Stalin"))))
```

- or, either are like and

Coordination

- We invited the psychics, but they were busy.

```
(c / contrast-01
   :ARG1 (i / invite-01
            :ARG0 (w / we)
            :ARG1 (p / person :mod (p2 / psychic)))
   :ARG2 (b / busy
            :domain p))
```

- We invited the psychics: they were busy. (*implicit*)
  - When in doubt, use and for implicit connectives.

Sentence Coordination

- *We invited the psychics.* But they were busy.

```
(c / contrast-01
  :ARG2 (b / busy
            :domain (t / they)))
```

Coordination: shared core args

- We invited and then disinvited the psychics.

```
(a / and
   :ARG1 (i / invite-01
            :ARG0 (w / we)
            :ARG1 (p / person :mod (p2 / psychic)))
   :ARG2 (d / disinvite-01
            :ARG0 w
            :ARG1 p
            :time (t / then)))
```

Coordination: shared non-core args

- Yesterday we invited and then disinvited the psychics.

```
(a / and
   :ARG1 (i / invite-01
            :ARG0 (w / we)
            :ARG1 (p / person :mod (p2 / psychic)))
   :ARG2 (d / disinvite-01
            :ARG0 w
            :ARG1 p
            :time (t / then))
   :time (y / yesterday))
```

Condition

- If we invite the psychics, they will attend.

```
(a / attend-01
   :ARG1 (p / person :mod (p2 / psychic)))
   :condition (i / invite-01
                 :ARG0 (w / we)
                 :ARG1 p))
```

- They won't attend unless we invite them.
  (They won't attend if we don't invite them.)

```
(a / attend-01 :polarity -
   :ARG1 (t / they))
   :condition (i / invite-01 :polarity -
                 :ARG0 (w / we)
                 :ARG1 t))
```

Concession

- Although we invited the psychics, they did not attend.

```
(a / attend-01
   :ARG1 (p / person :mod (p2 / psychic)))
   :concession (i / invite-01
                  :ARG0 (w / we)
                  :ARG1 p))
```

- The psychics did not attend, in spite of our invitation.

Cause

- The psychics attended because we invited them.

```
(a / attend-01
   :ARG1 (p / person :mod (p2 / psychic)))
   :cause (i / invite-01
             :ARG0 (w / we)
             :ARG1 p))
```

- The psychics' attendance was due to our invitation.

Purpose

- We invited the psychics so that they would attend.

```
(i / invite-01
   :ARG0 (w / we)
   :ARG1 (p / person :mod (p2 / psychic))
   :purpose (a / attend-01
               :ARG1 p))
```

Review


tails
Complimentary
EAT, RAY, LOVE
Rachael Ray finds inspiration in cooking her family and her dog
10 TIP
TO KEEP YO
PET HEALTH
AND HAPP

Review

- Rachael Ray finds inspiration in cooking her family and her dog.

```
(i / inspire-01
   :ARG0 (c / cook-01
                                                  )
   :ARG1 (p / person :name (n / name :op1 "Rachael" :op2 "Ray")))
```

Review

- Rachael Ray finds inspiration in cooking her family and her dog.

```
(i / inspire-01
   :ARG0 (c / cook-01
            :ARG0 p
            :ARG1 (a / and
                     :op1 (f / family
                                                              )
                     :op2 (d / dog          )))
   :ARG1 (p / person :name (n / name :op1 "Rachael" :op2 "Ray")))
```

Review

- Rachael Ray finds inspiration in cooking her family and her dog.

```
(i / inspire-01
   :ARG0 (c / cook-01
            :ARG0 p
            :ARG1 (a / and
                     :op1 (f / family
                                                              )
                     :op2 (d / dog :poss p)))
   :ARG1 (p / person :name (n / name :op1 "Rachael" :op2 "Ray")))
```

Review

- Rachael Ray finds inspiration in cooking her family and her dog.

```
(i / inspire-01
   :ARG0 (c / cook-01
            :ARG0 p
            :ARG1 (a / and
                     :op1 (f / family
                              :ARG1-of (h / have-org-role-91
                                          :ARG0 p
                                          :ARG2 (m / member)))
                     :op2 (d / dog :poss p)))
   :ARG1 (p / person :name (n / name :op1 "Rachael" :op2 "Ray")))
```

X's family = family of which X is a member

Review

- Rachael Ray finds inspiration in cooking, her family, and her dog.

```
(i / inspire-01
   :ARG0 (a / and
            :op1 (c / cook-01
                    :ARG0 p)
            :op2 (f / family
                    :ARG1-of (h / have-org-role-91
                                :ARG0 p
                                :ARG2 (m / member)))
            :op3 (d / dog :poss p))
   :ARG1 (p / person :name (n / name :op1 "Rachael" :op2 "Ray")))
```

AMR

Modality, Mood, & Speech Acts

Negation

- no worries

```
(w / worry-01
    :polarity -)
```

Imperative

- Don’t worry!

```
(w / worry-01
    :mode imperative
    :ARG0 (y / you)
    :polarity -)
```

Politeness

- Please don’t worry.

```
(w / worry-01
    :mode imperative
    :ARG0 (y / you)
    :polarity -
    :polite +)
```

Vocative

- Don’t worry, Simba. **TODO: coref with vocative & implied “you”?**

```
(s / say-01
   :ARG2 (l / lion :name (n / name :op1 "Simba"))
   :ARG1 (w / worry-01
            :mode imperative
            :ARG0 (y / you)
            :polarity -))
```

Quoted Speech

- `say-01` is the default for speech if no overt verb (incl. vocatives, “according to”, quotation marks)

- Pronouns in direct quotes are replaced with their antecedents if possible

Expressive

- Oy!

```
(o / oy
   :mode expressive)
```

- Limited to isolated interjections (and similar utterances lacking a predication)

Questions: yes-no

- Are you worried?

```
(w / worry-01
   :ARG0 (y / you)
   :mode interrogative)
```

- To be, or not to be?

```
(o / or
   :op1 (e / exist-01)
   :op2 (e2 / exist-01 :polarity -)
   :mode interrogative)
```

Questions: wh

- Why worry? (What is the point of worrying?)

```
(w / worry-01
   :ARG0 (y / you)
   :purpose (a / amr-unknown))
```

Think of `amr-unknown` as an *in situ* question pronoun. Structurally, the AMR is the same as a declarative sentence.

- What's the problem?

```
(p / problem
   :domain (a / amr-unknown))
```

- How many peppers did Peter Piper pick?

```
(p / pick-10
   :ARG0 (p2 / person :name (n / name :op1 "Peter" :op2 "Piper"))
   :ARG1 (p3 / pepper
             :quant (a / amr-unknown)))
```

Modal Concepts

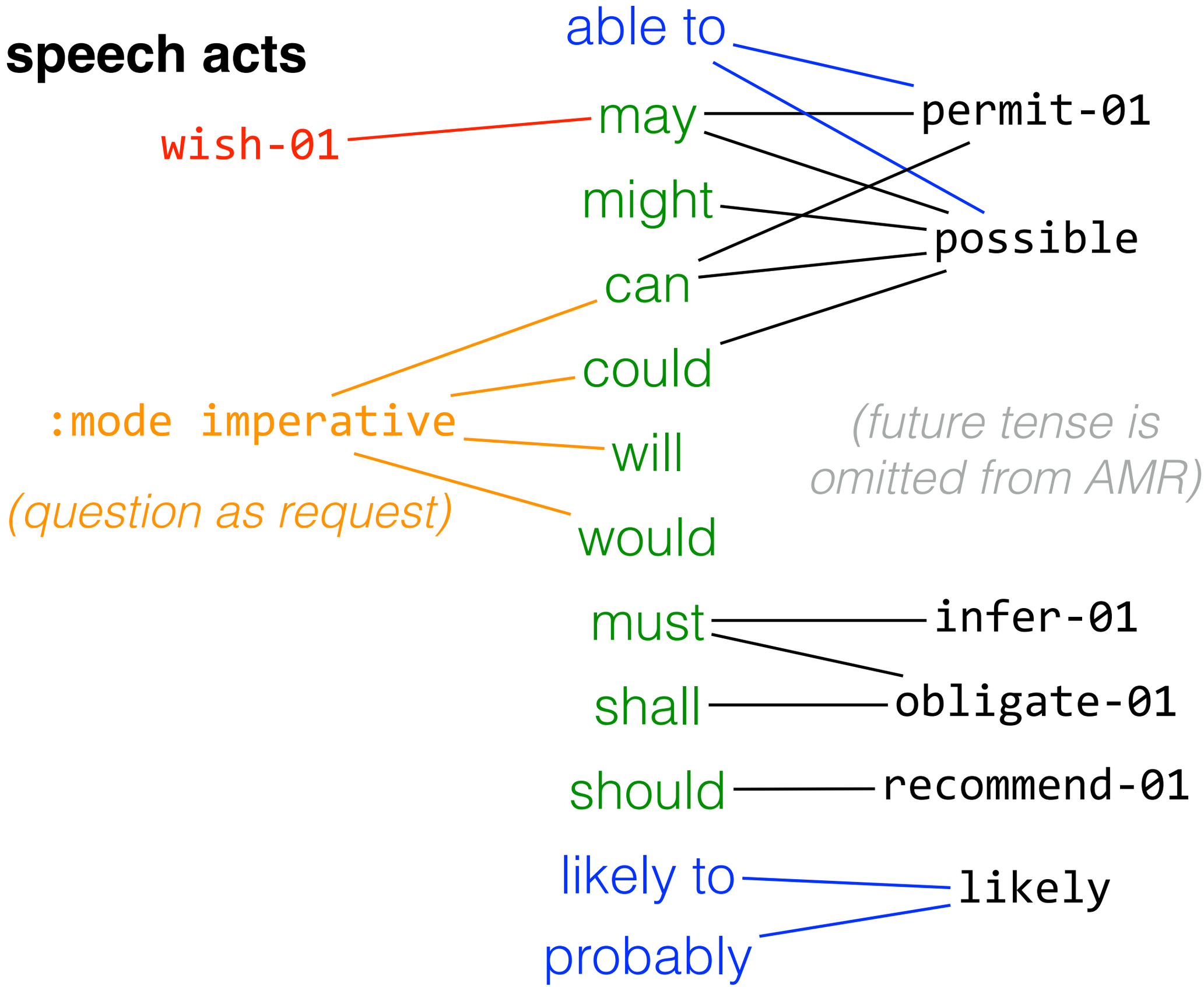
- You can leave.
  You may leave.
  It's all right for you to leave.

```
(p / permit-01
   :ARG1 (l / leave-01
            :ARG0 (y / you))
```

- I can see Russia from my house!
  I'm able to see Russia from my house!

```
(p / possible
   :domain (s / see-01
              :ARG0 (i / i)
              :ARG1 (c / country :name (n / name :op1 "Russia"))
              :location (h / house :poss i)))
```

speech acts


able to
may
might
can
could
will
would
must
shall
should
likely to
probably
wish-01
permit-01
possible
infer-01
obligate-01
recommend-01
likely
:mode imperative
(question as request)
(future tense is omitted from AMR)

AMR

Quantification & Comparison

Quantification

- two apples

```
(a / apple
   :quant 2)
```

- a lot of apples

```
(a / apple
   :quant (l / lot))
```

- All apples are fruit.

```
(f / fruit
   :domain (a / apple
              :quant (a / all)))
```

Only explicit quantifiers are included in the AMR.

- Apples are fruit.

```
(f / fruit
   :domain (a / apple))
```

Quantification

- two apples or: **There are** 2 apples.

```
(a / apple
   :quant 2)
```

- a lot of apples or: **There are** lots of apples.

```
(a / apple
   :quant (l / lot))
```

Existentials are not represented in the AMR.

- All apples are fruit.

```
(f / fruit
   :domain (a / apple
              :quant (a / all)))
```

- Apples are fruit.

```
(f / fruit
   :domain (a / apple))
```

Comparison

- The current treatment of comparative constructions is very shallow. (We are working on improvements.)

- *(I ate)* more apples than bananas

```
(a / apple
   :quant (q / quantity
             :degree (m / more)
             :compared-to (b / banana)))
```

- *(I ate)* as many apples as bananas

```
(a / apple
   :quant (q / quantity
             :degree (e / equal)
             :compared-to (b / banana)))
```

Many other kinds of “as…as” are documented in the AMR Dictionary

Comparison

- Apples are redder than bananas.

```
(r / red
   :domain (a / apple)
   :degree (m / more)
   :compared-to (b / banana))
```

- Apples are redder than bananas are yellow.

```
(r / red
   :domain (a / apple)
   :degree (m / more)
   :compared-to (y / yellow
                   :domain (b / banana)))
```

Comparison

- I like apples less than bananas.

```
(l / like-01
   :ARG0 (i / i)
   :ARG1 (a / apple)
   :degree (l2 / less)
   :compared-to (b / banana)))
```

- I like apples less than I like bananas.

```
(l / like-01
   :ARG0 (i / i)
   :ARG1 (a / apple)
   :degree (l2 / less)
   :compared-to (l3 / like-01
                      :ARG0 i
                      :ARG1 (b / banana)))
```

Comparison

- I like apples more than Johnny Appleseed.

```
(l / like-01
   :ARG0 (i / i)
   :ARG1 (a / apple)
   :degree (m / more)
   :compared-to (p / person :name (n / :op1 "Johnny" :op2 "Appleseed")))
```

- I like apples more than Johnny Appleseed does.

```
(l / like-01
   :ARG0 (i / i)
   :ARG1 (a / apple)
   :degree (m / more)
   :compared-to (l2 / like-01
                    :ARG0 (p / person
                               :name (n / :op1 "Johnny" :op2 "Appleseed"))
                    :ARG1 a))
```